system cannot load all required data and data synchronization becomes a potential problem. Prior research into multi-threaded rule engines were unsuccessful because of the cost of maintaining locks. Distributed reasoning does not use locking mechanism and does not suffer from synchronization issues.

What is claimed is:

[Claim 1] (currently amended) A rule engine which implements the RETE algorithm with novel extensions, which can distribute patterns defined by a set of rules.

[Claim 2] (currently amended) A system according to claim 1 converts an object structure to an internal structure, which the rule engine uses to perform operations and reason over data.

[Claim 3] (currently amended) A system according to claim 1stores data in a working memory; which may span multiple partitions across several computer systems and is capable of distributing indexes across multiple rule engines.

[Claim 4] (currently amended) A system according to claim 1 creates an input nodes for each class type and is capable of propagating data through a local network, or remotely to a rule engine running on a separate computer system.

[Claim 5] (currently amended) A system according to claim 1 uses a 2 input Join node to evaluate 2 or more data elements and determines if the data should propagate further through the network on a separate system or locally.

[Claim 6] (currently amended) A system according to claim 1 uses a Terminal nodes to indicate all the conditions of a rule have been met and is a candidate for immediate or delayed execution locally or on a remote computer system.

[Claim 7] (currently amended) A pattern for a given class type according to claim 2 uses a linear sequence of nodes representing the patterns to match for the class.

[Claim 8] (currently amended) A system according to claim 1 implements an abstraction layer for retrieving remote facts residing in a separate rule engine process.

[Claim 9] (original) A system according to claim 1 uses a call back mechanism between the working memory, the rule engine and the objects; which objects use to notify the rule engine of changes.

[Claim 10] (currently amended) A system according to claim 1 requires object instantiations to implement a defined interface for the call back mechanism, which may be added to the classes using byte code manipulation.

[Claim 11] (currently amended) A system according to claim 1 monitors the resource usage; which is used to determine when RETE nodes are distribute to other rule engines.

[Claim 12] (original) A system according to claim 1 uses rules to manage the distribution of nodes to remote systems.

[Claim 13] (currently amended) A system according to claim 1 distributes pattern matching by copying the nodes and sending them to a remote system; which are then added to remote engine's network.

[Claim 14] (currently amended) A system according to claim 1 distributes the class type node, 2 input node, terminal node and intra-element nodes to a remote system.

[Claim 15] (original) Distributed nodes according to claim 14 maintains a list of remote systems which depend on the indexes produced by pattern matching.

[Claim 16] (currently amended) A system according to claim 1 will serialize the values of an object to a remote system if the corresponding pattern matches against a remote object pattern, or it may perform join using the indexes directly.

[Claim 17] (currently amended) A system according to claim 1 may serialize the data object and its values to a remote system, if the object contains procedural logic and functional attachments.

[Claim 18] (original) A system according to claim 1 may serialize the values of an object to a remote system and the receiving system may create a new instance of the object for pattern matching.

[Claim 19] (original) Objects according to claims 16 to 18 are considered temporal by the rule engine if the object's original instance and nodes reside on a remote system.

[Claim 20] (original) A system according to claim 1 defines three types of input channels: standard input, data distribution and node distribution.

[Claim 21] (currently amended) A system according to claim 1 defines a set of application interfaces and components to handle incoming data input, events and requests for pattern matching.

[Claim 22] (original) Input according to claim 21 is defined as standard input channel.

[Claim 23] (original) A system according to claim 1 defines a data distribution channel for sending and receiving remote data between rule engines.

[Claim 24] (currently amended) A system according to claim 1 defines a pattern distribution channel for distributing RETE nodes and patterns.

[Claim 25] (original) A system according to claim 1 considers an object as temporal if it was sent through the data distribution channels.

[Claim 26] (currently amended) Temporal objects according to claim 19 and 25 are used by the engine to perform pattern matching and may be discarded immediately after the pattern matching process is complete.

[Claim 27] (currently amended) A system according to claim 1 will route the index results of claim 26 back to the originating system using the data distribution channel.

[Claim 28] (original) A system according to claim 1 will update the index of the join and terminal nodes as a result of pattern matching according to claim 26.

[Claim 29] (original) A system according to claim 1 processes the outputs of the rule and produces a set of results, if the original event began locally.

[Claim 30] (original) A system according to claim 1 uses messaging system to route new events to a cluster of rule engines.

[Claim 31] (original) A messaging system according to claim 30 filters new messages and routes them to the correct engine.

[Claim 32] (original) A system according to claim 1 uses messaging system to route the final results as defined by a rule to the recipient.

[Claim 33] (original) A system according to claim 1 contains a component responsible for communicating with the messaging system.

[Claim 34] (original) A messaging component according to claim 33 is responsible for processing inbound events and generating new messages for outbound publication.

[Claim 35] (currently amended) A system according to claim 1 prefers to process new data events asynchronously using a messaging system, but is not required to use asynchronous communication.

[Claim 36] (original) Distributed nodes according to claim 14 contain information about the originating engine, a timestamp of when the nodes were distributed and a priority attribute.

[Claim 37] (original) The priority attribute according to claim 36 may be used by the engine to remove the nodes, if all local object instances have been removed from the working memory.

[Claim 38] (currently amended) Distributed nodes according to claim 14 will not be removed from the local pattern matching network, if data for those patterns is still being used, either in active rules about to fire or in remote rule engine instances.

[Claim 39] (original) A system according to claim 1 may forward node distribution messages, if the system does not have sufficient resources.

[Claim 40] (original) A forward message according to claim 19 must retain the location of the originating engine and add the current engine's unique runtime name to a list of recipients.